A Genetic Algorithm for Learning Parameters in Bayesian Networks using Expectation Maximization

Priya K. Sundararajan & Ole J. Mengshoel
Carnegie Mellon University, Silicon Valley

International Conference on Probabilistic Graphical Models
Lugano, Switzerland, September 6-9, 2016

Outline

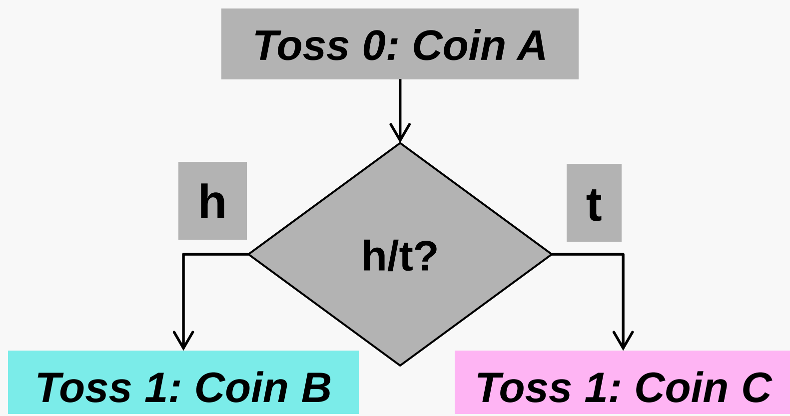
- Expectation Maximization (EM) Review
- Challenges of EM and Current EM Approaches
- Our GAEM Approach
- GAEM Replacement Methods
- Experimental Results
- Discussion and Future Work

Outline

- Expectation Maximization (EM) Review
- Challenges of EM and Current EM Approaches
- Our GAEM Approach
- GAEM Replacement Methods
- Experimental Results
- Discussion and Future Work

Maximum Likelihood Estimation
Complete Data

Coin Toss Experiment



Data

t	*hhh*
h	*ttt*
t	*hhh*
h	*ttt*

Model

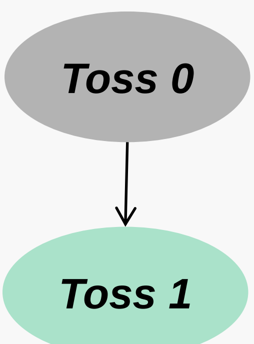
$\theta = (\lambda, p_1, p_2)$

λ = *P(Coin A = h)*

p_1 = *P(Coin B = h)*

p_2 = *P(Coin C = h)*

Bayesian Network



h	*0.5*
t	*0.5*

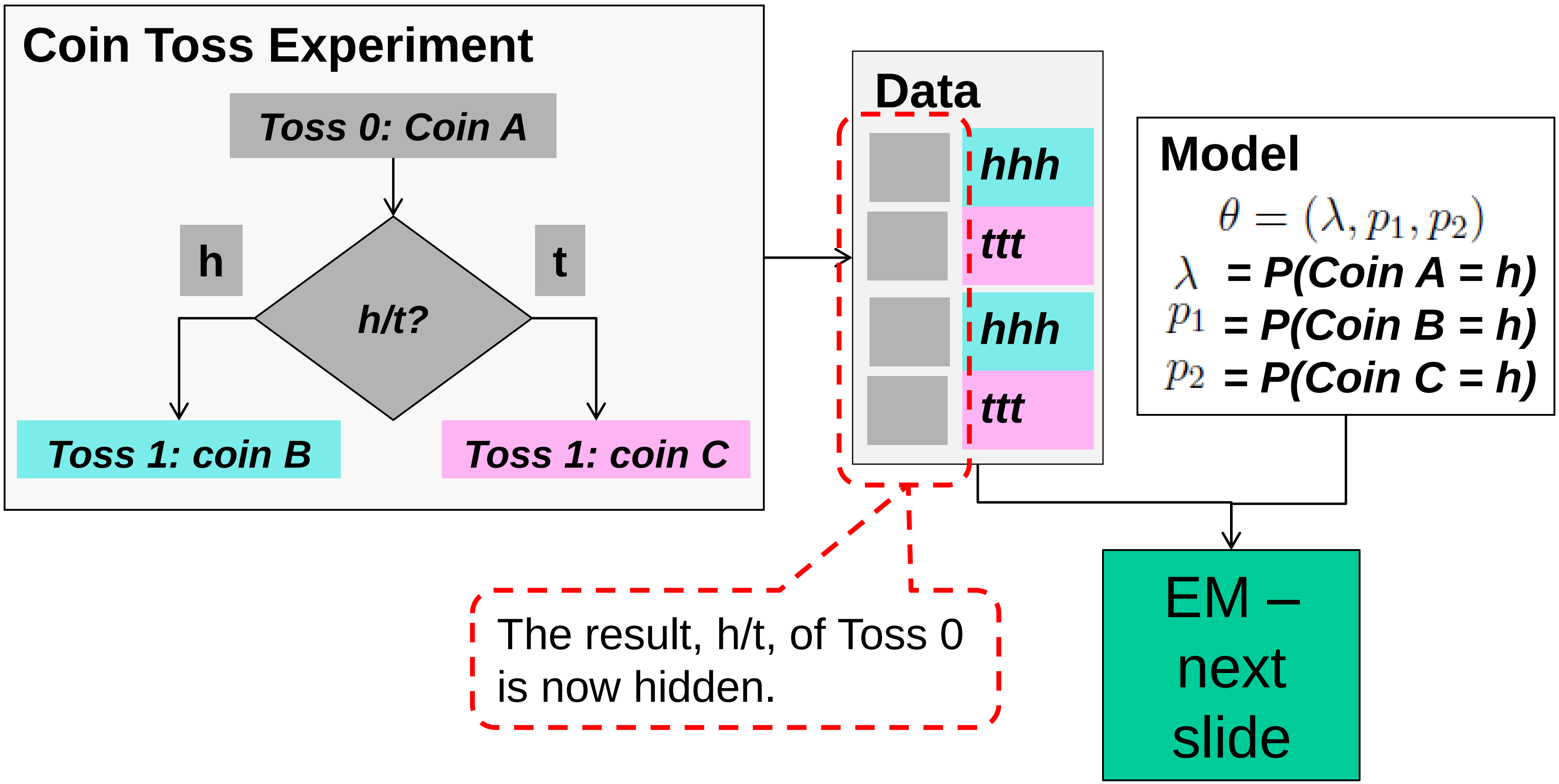
Coin A	*h*	*t*
	Coin B	*Coin C*
h	*1*	*0*
t	*0*	*1*

MLE

$$\lambda = \frac{1}{2} \qquad p_1 = \frac{6}{6} \qquad p_2 = \frac{0}{6}$$

Ref: http://www.cs.columbia.edu/~mcollins/6864/slides/em1.4up.pdf

Expectation Maximization (EM)
Incomplete Data



EM - Three Coin Tossing Experiment

Random Parameters

$$\lambda^{(0)} = 0.3$$
$$p_1^{(0)} = 0.3$$
$$p_2^{(0)} = 0.6$$

E-Step: Estimate Parameters

Data	Coin A	Coin B	Coin C
hhh	0.0508	0.15h 0t	2.8h 0t
ttt	0.6967	0h 2.2t	0h 0.9t
hhh	0.0508	0.15h 0t	2.8h 0t
ttt	0.6967	0h 2.2t	0h 0.9t
	1.495	0.3h 4.4t	5.6h 1.8t

M-Step: Update Parameters

$$\lambda^{(1)} = \frac{1.495}{4} = 0.3738$$
$$p_1^{(1)} = \frac{0.3048}{4.485} = 0.0681$$
$$p_2^{(1)} = \frac{5.695}{7.515} = 0.7578$$

Converged?

No

Yes

Converged Parameters

$$\lambda^{(t)} = 0.5$$
$$p_1^{(t)} = 0.0$$
$$p_2^{(t)} = 1.0$$

EM generally converges , in a hill-climbing fashion, to a local maximum of the (log-)likelihood.

From Complete to Incomplete Data

- Complete Data
  - Let $x = (x_1, x_2, \dots x_n)$ be a data vector and ω be the parameter.
  - Probability of the data: $P(x|\omega) = P(x_1|\omega)\, P(x_2|\omega) \dots P(x_n|\omega)$.
  - Likelihood function: $L(\omega|x) = P(x|\omega) = \prod_{i=1}^{n} P(\, x_i|\omega)$.
  - Log-likelihood (LL): $l(\omega|x) = \sum_{i=1}^{n} \log P(x_i|\omega)$.
  - Maximum Likelihood Estimation (MLE): $\omega_{ML} = argmax_{\omega} l(\omega|x)$.
- Incomplete Data
  - Let $y = (y_1, y_2, \dots, y_m)$ be the missing data.
  - Log-likelihood (LL): $l(\omega|x, y) = \sum_{i=1}^{n} \log \sum_{j=1}^{m} P(x_i, y_j|\omega)$.
  - Expectation Maximization: $\omega_{EM} = argmax_{\omega} l(\omega|x, y)$.

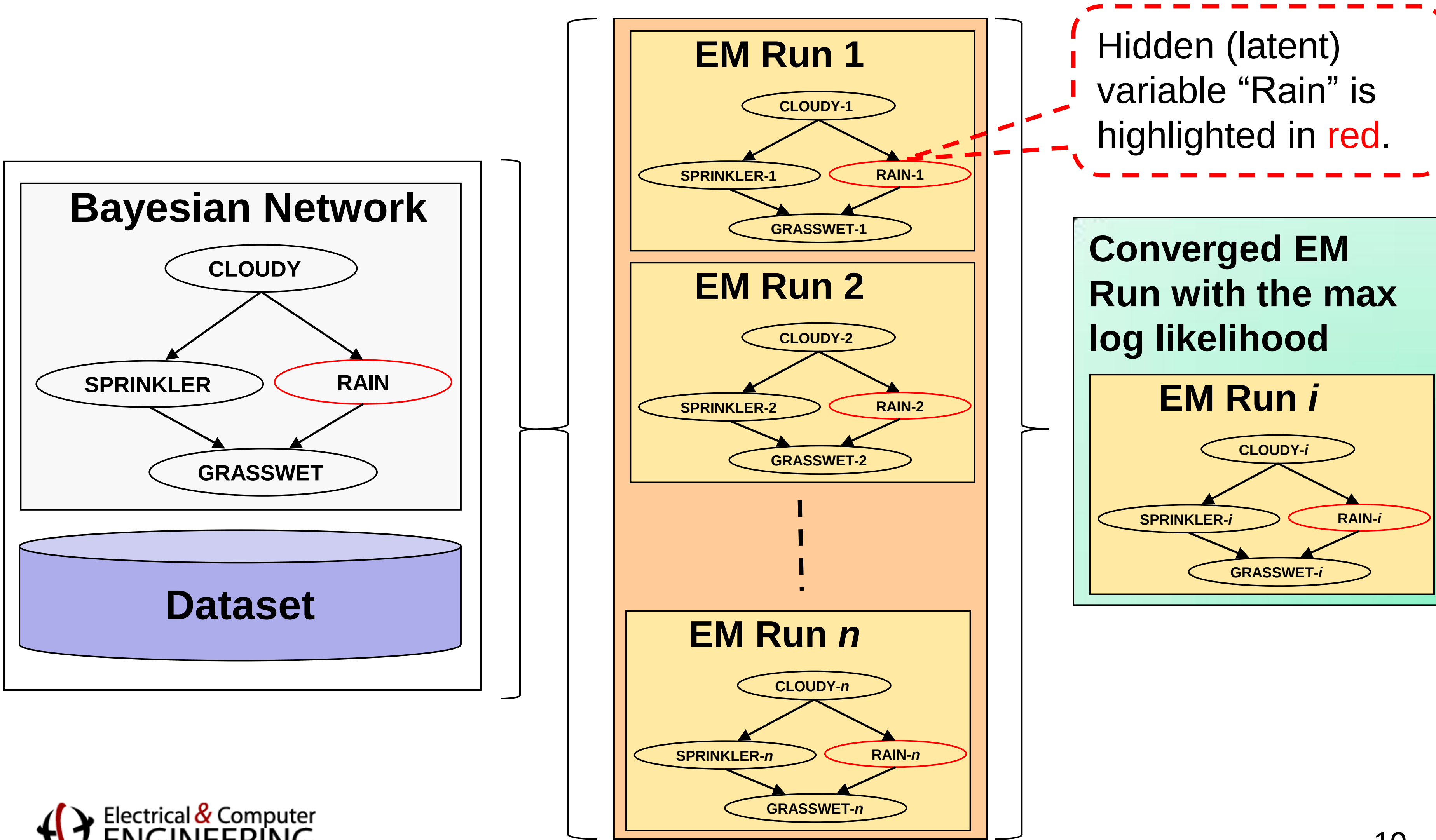
Outline

- Expectation Maximization (EM) Review
- Challenges of EM and Current EM Approaches
- Our GAEM Approach
- GAEM Replacement Methods
- Experimental Results
- Discussion and Future Work

Challenges of EM

- Problem of local maxima – multimodal search space
- Problem of slow convergence – many EM iterations
- Problem of computational complexity of E-Step (and M-Step)

Traditional EM: Multiple Random Starting Points Strategy



Outline

- Expectation Maximization (EM) Review
- Challenges of EM and Current EM Approaches
- Our GAEM Approach
- GAEM Replacement Methods
- Experimental Results
- Discussion and Future Work

Genetic Algorithm for Expectation Maximization (GAEM)

GAEM's goal is to speed up and improve LL, specifically to …

- Improve handling of local maxima – randomness of GA helps to escape local maxima and
- Improve robustness to poor initialization – fitter learned individuals are used as parents for next generation

… by combining

- The monotonic improvement property of EM and
- The stochastic property of GA

GAEM: Integrating GA and EM

- Representation – Each GA individual encodes the parameters of a Bayesian network
- Parameters – Genes
- Bag of individuals – Population
- Recombination of *c = 2* individuals:

$$\theta^{a} = (\theta_{a1}, \theta_{a2}, \theta_{a3}, \theta_{a4}, \theta_{a5}, \theta_{a6})$$

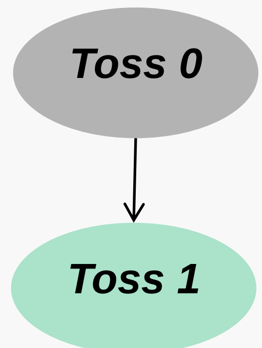
$$\theta^{b} = (\theta_{b1}, \theta_{b2}, \theta_{b3}, \theta_{b4}, \theta_{b5,}, \theta_{b6})$$

  - After Crossover

$$\theta_c^{a} = (\theta_{a1}, \theta_{a2}, \theta_{b3}, \theta_{b4}, \theta_{b5}, \theta_{b6})$$

$$\theta_c^{b} = (\theta_{b1}, \theta_{b2}, \theta_{a3}, \theta_{a4}, \theta_{a5}, \theta_{a6})$$

Bayesian Network

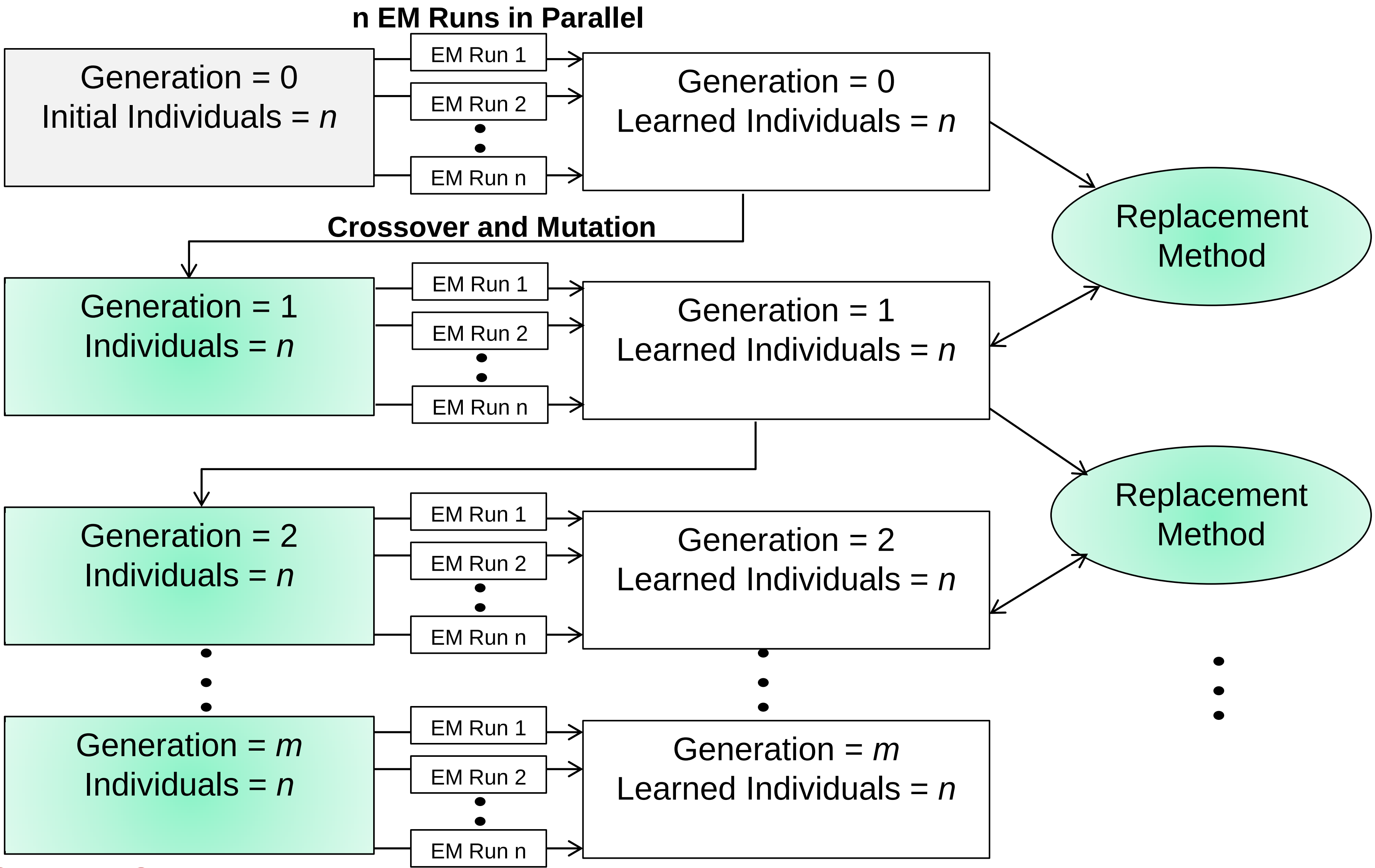


h	0.5
t	0.5

Coin A	h	t
	Coin B	Coin C
h	1	0
t	0	1

- Mutation of one individual: $\theta_m^{a} = (\theta_{a1}, \theta_{a2}, \theta'_{b3}, \theta_{b4}, \theta_{b5}, \theta_{b6})$
- Replacement – Based on fitness
- Fitness function – Log-likelihood (LL) value

GAEM: Behavior over the Generations



Outline

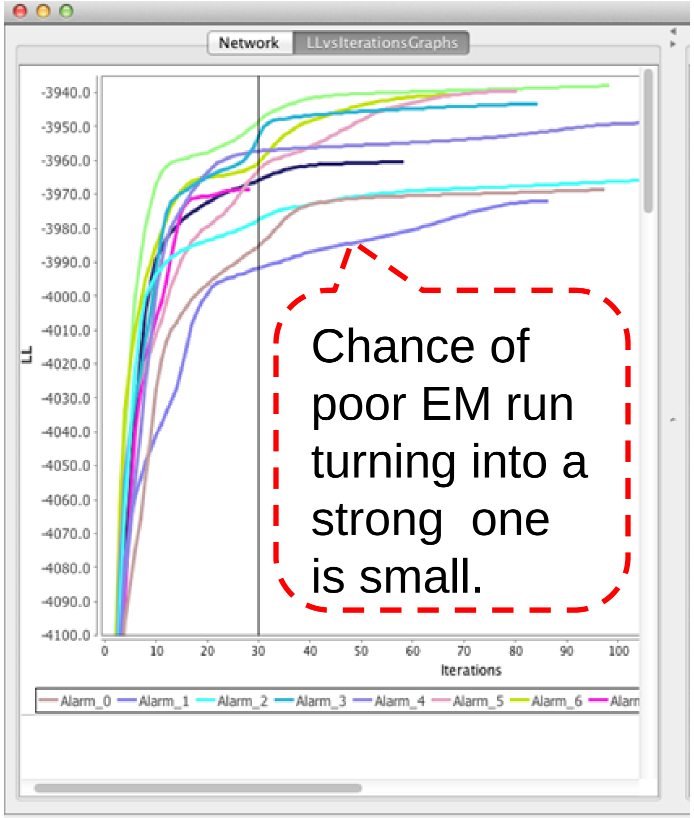
- Expectation Maximization (EM) Review
- Challenges of EM and Current EM Approaches
- Our GAEM Approach
- GAEM Replacement Methods
- Experimental Results
- Discussion and Future Work

Four GAEM Replacement Methods

1. Direct replacement (GAEM-TRAD)
   - *If (f(parent1) > f(child1))* ? *parent1* : *child1*
2. Deterministic Crowding (GAEM-DETER)
   - Find distances of parent and child using KL divergence, use them to
     - *val1 = d(parent1,child1) + d(parent2,child2)*
     - *val2 = d(parent1,child2) + d(parent2,child1)*
   - *If (val1 < val2) ? compare(parent1, child1), compare(parent2, child2) : compare(parent1, child2), compare(parent2, child1)*
3. Probabilistic Crowding(GAEM-PC)
   - *P(parent1) = f(parent1)/(f(parent1) + f(child1))*
   - *parent1* wins with probability *P(parent1)*
4. ALEM based replacement (GAEM-ALEM): Next slide

GAEM: ALEM-Based Replacement

(1) Traditional EM



(2) Intuition for GAEM-ALEM:

- *P(Poor → Strong EM Run)* is low
- Discard Poor EM Runs
- Save CPU cycles

(3) Pseudo-code for GAEM-ALEM:

For each generation

- After *n* EM iterations, compare child with parent EM run: *if (f(parent) > f(child)) ? parent : child*

Outline

- Expectation Maximization (EM) Review
- Challenges of EM and Current EM Approaches
- Our GAEM Approach
- GAEM Replacement Methods
- Experimental Results
- Discussion and Future Work

Experimental Setup – GAEM Parameters

GA Parameters	Values
Population size (n_p)	2, 4, 8, 16 and 32
Genes per individual	Alarm BN = 37, Carstarts BN = 18, Hepar2 BN = 70, Win95pts BN = 76, Child BN = 20, Hailfinder BN = 56, Insurance BN = 27, Sprinkler BN = 4
Mating	Random
Crossover probability (p_c)	Hard Search Space: p_c= 0.1 (single point crossover) Easy Search Space: p_c = 0.5 (single point crossover)
Mutation probability (p_m)	Hard Search Space: p_m= 0.1 Easy Search Space: p_m= 0.05
Replacement (α)	GAEM-TRAD, GAEM-DETER, GAEM-PC, GAEM-ALEM
GA type	Generational
Number of generations (n_g)	10

Experimental Setup – BNs, HW, and SW

Bayesian Network Name	Number of nodes	Number of hidden (latent) variables
Child	20	10
Insurance	27	13
Sprinkler	4	2
Carstarts	18	7
Alarm	37	19
Hepar2	70	35
Win95pts	76	38
Hailfinder	56	28

Hardware used:

Processor : Intel Xeon
Memory(RAM) : 24GB
CPU : 2.4 GHz 16 core

Software used:

Library : libDAI[1]
Multithreading : Boost
Language used : C++ and shell scripts
OS : Linux

Sample sizes used:

500, 1000, 1500, 2000, 2500 and 3000

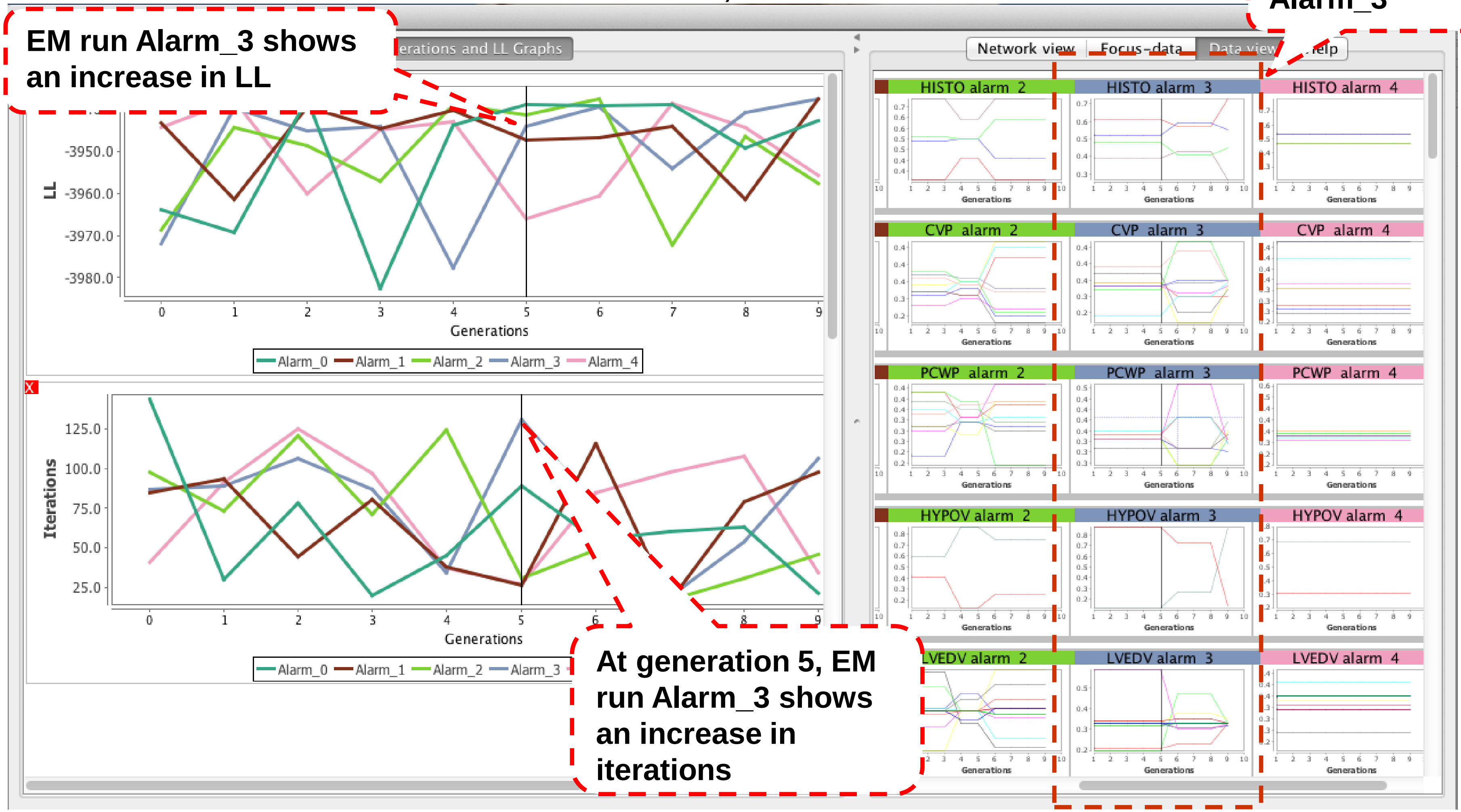
[1] *libDAI: http://cs.ru.nl/~jorism/libDAI/*

Visualization – EM Learning (GAEM)

Pm = 0.9; Pc = 0.5

One EM Run: Alarm_3

EM run Alarm_3 shows an increase in LL



At generation 5, EM run Alarm_3 shows an increase in iterations

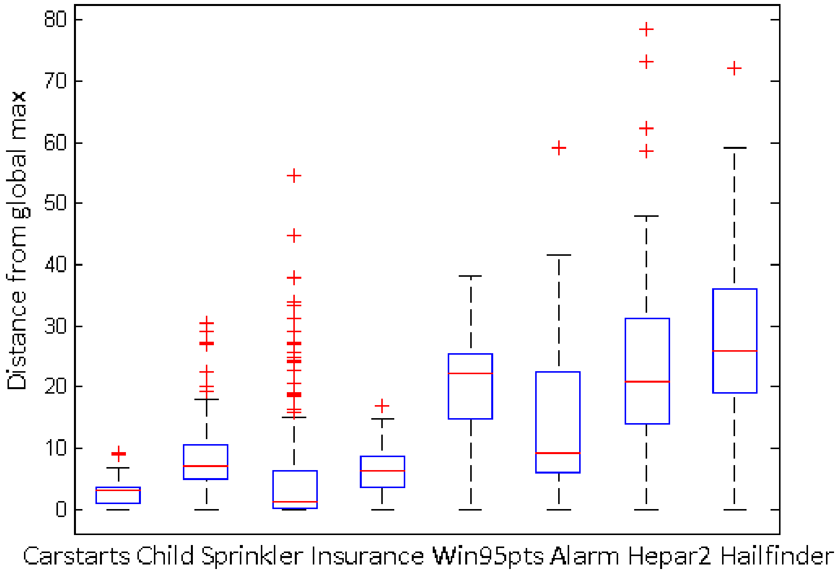
Experiment 1: How do we Characterize EM Search Spaces?

- Bayesian networks used:
  - Carstarts, Child, Sprinkler, Insurance, Win95pts, Alarm, Hepar2 and Hailfinder.
- For each Bayesian network 200 EM Runs are generated.
- Sample size: 500.
- Traditional EM algorithm is run until convergence.
- Distance from the best log likelihood is calculated:

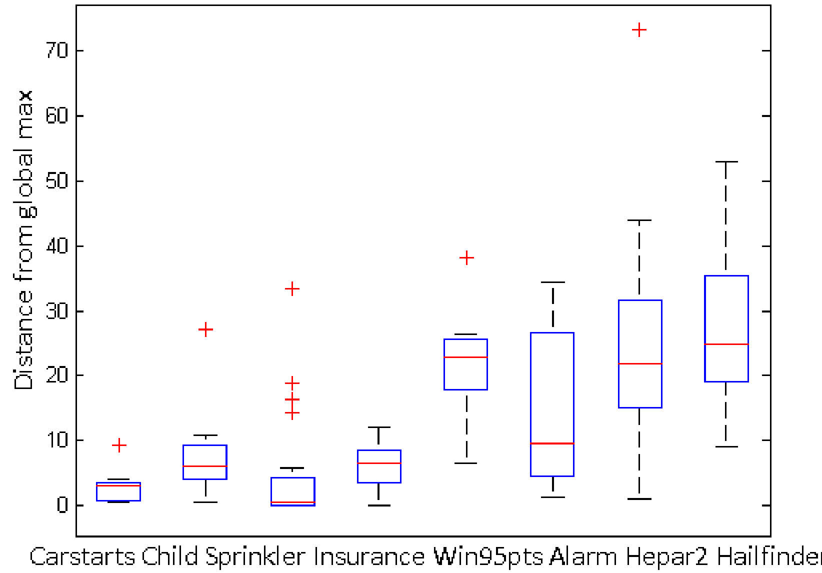
$$d_i = l^* - l_i.$$

Experiment 1: Search Space Analysis

200 Traditional EM runs



20 Traditional EM runs

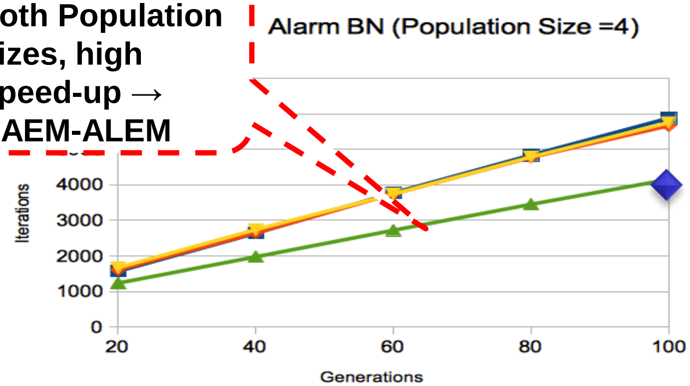


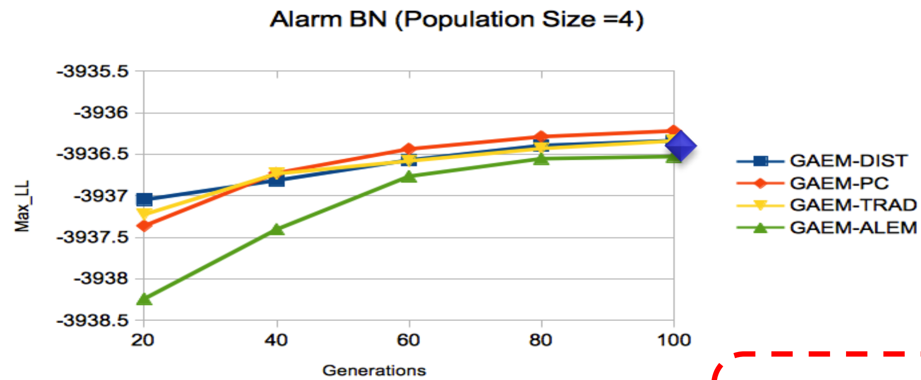
- Easy search spaces: Median is close to the global max (Carstarts, Child, Sprinkler and Insurance). In Win95pts, 50% of EM runs above median show less spread.
- Hard search spaces: Spread above median is high. 50% of EM runs are away from global max (Alarm, Hepar2 and Hailfinder).

Experiment 2: Effect of Replacement

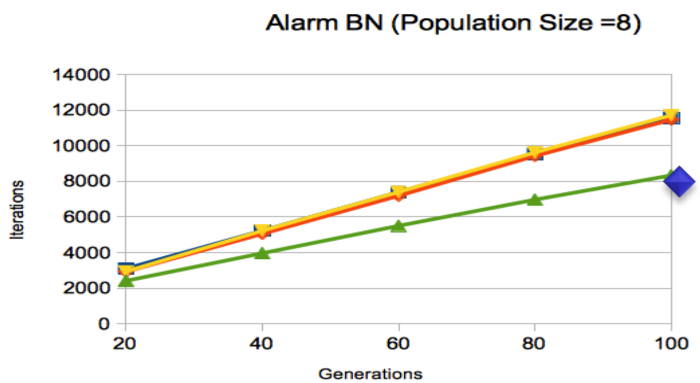
Generations = 100; Pm = 0.1; Pc = 0.1; Population Size = 4, 8

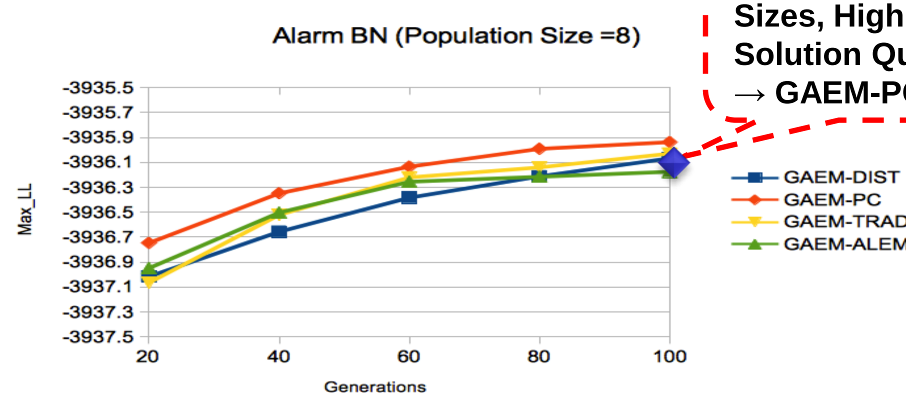
Both Population Sizes, high Speed-up → GAEM-ALEM





Both Population Sizes, High Solution Quality → GAEM-PC





Hard search space: GAEM-PC based replacement produces a high solution quality and GAEM-ALEM produces a high speed up for the Alarm BN.

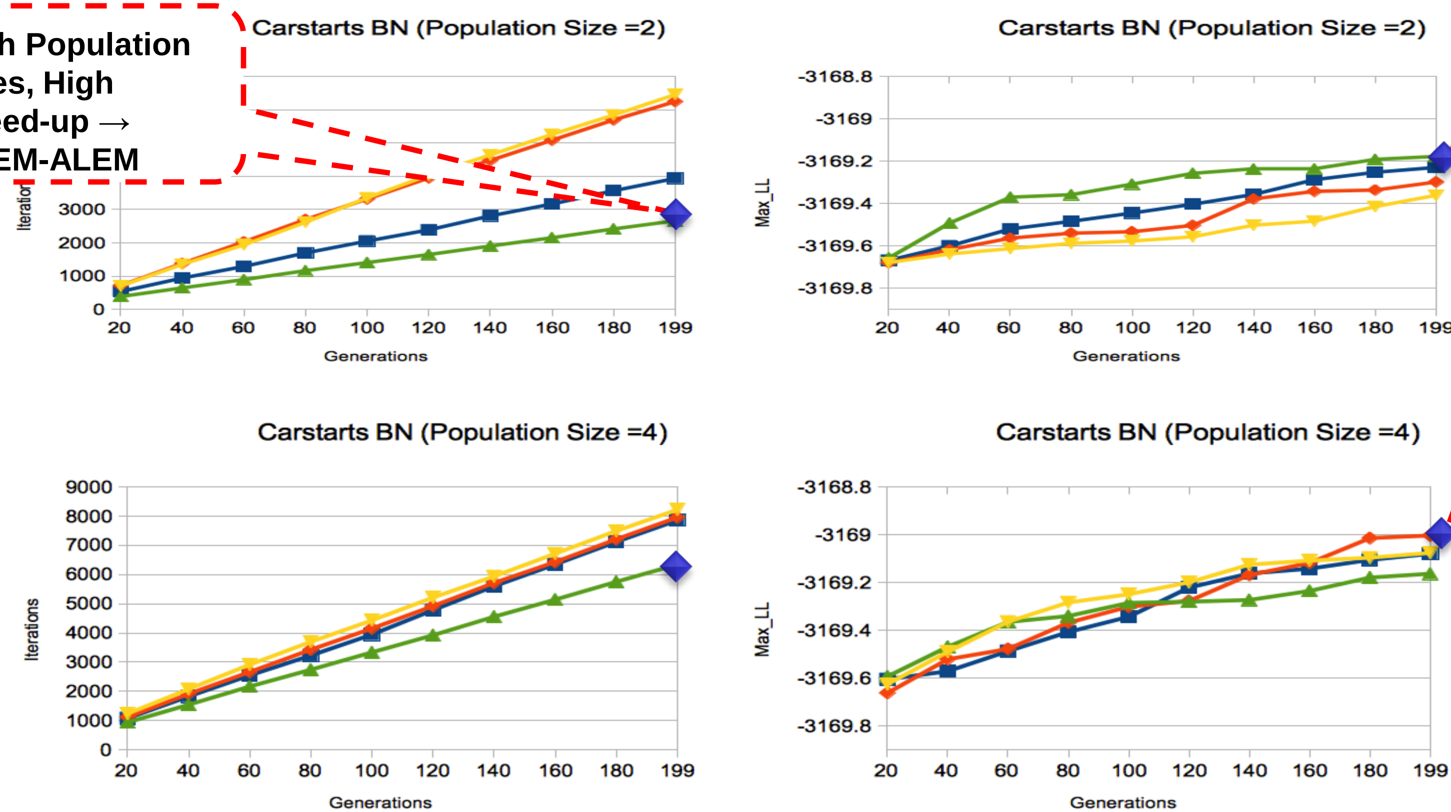
Experiment 2: Effect of Replacement

Generations = 200; Pm = 0.05; Pc=0.5; Population Size = 2, 4

Both Population Sizes, High Speed-up → GAEM-ALEM

Population Size =2, High Solution Quality → GAEM-ALEM

Population Size =4, High Solution Quality → GAEM-PC



Easy search space: For small population, GAEM-ALEM produces a high solution quality and speed-up. GAEM-PC gives higher solution quality as population size is increased.

Experiment 3: Speed-Up Results

Carstarts BN : Pm = 0.1; Pc=0.1; Population Size = 2; Generations = 200
Alarm BN : Pm = 0.05; Pc=0.5; Population Size = 4; Generations =100

GAEM solution quality is generally higher than traditional EM:

	Carstarts BN				Alarm BN			
Samples n_s	GAEM-TRAD	GAEM-PC	GAEM-ALEM	EM	GAEM-TRAD	GAEM-PC	GAEM-ALEM	EM
500	-3169.40	-3169.55	**-3169.31**	-3169.96	-3936.93	**-3936.69**	-3937.31	-3937.44
1000	-6924.56	-6924.56	**-6924.54**	-6924.80	**-16047.15**	-16048.18	-16048.6	-16050.20
1500	**-8646.85**	**-8646.85**	**-8646.85**	**-8646.85**	-22977.85	-22977.89	**-22979.56**	-22981.7
2000	-13743.87	-13744.50	**-13743.84**	-13743.85	-31217.09	**-31217.02**	-31217.90	-31218.10
2500	-14504.44	-14504.46	-14504.43	**-14504.40**	**-40550.40**	-40550.97	-40552.73	-40556.00
3000	-18888.15	**-18887.61**	-18887.84	-18887.90	**-51210.45**	-51211.13	-51217.46	-51220.50

GAEM speed-up is1.5x to 7.0x:

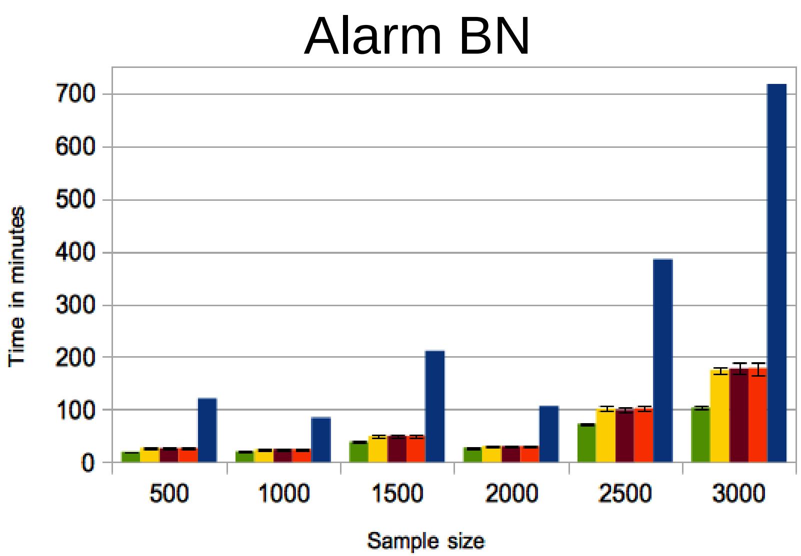
	Carstarts BN			Alarm BN		
Samples n_s	GAEM-TRAD	GAEM-PC	GAEM-ALEM	GAEM-TRAD	GAEM-PC	GAEM-ALEM
500	2049 (4.1)	2757 (3.1)	1397 (**6.0**)	3062 (4.6)	3172 (4.5)	2322 (**6.1**)
1000	1227 (4.6)	1765 (3.2)	1016 (**5.6**)	1659 (3.5)	1631 (3.6)	1386 (**4.2**)
1500	624 (**1.5**)	624 (**1.5**)	624 (**1.5**)	2515 (4.2)	2522 (4.2)	1940 (**5.5**)
2000	1282 (4.4)	1231 (4.6)	989 (**5.8**)	1097 (3.6)	1107 (3.6)	997 (**4.0**)
2500	688 (2.0)	683 (**2.1**)	684 (**2.1**)	2507 (3.8)	2499 (3.8)	1774 (**5.4**)
3000	1214 (5.2)	1208 (5.2)	977 (**6.5**)	4082 (4.0)	4002 (4.1)	2353 (**7.0**)

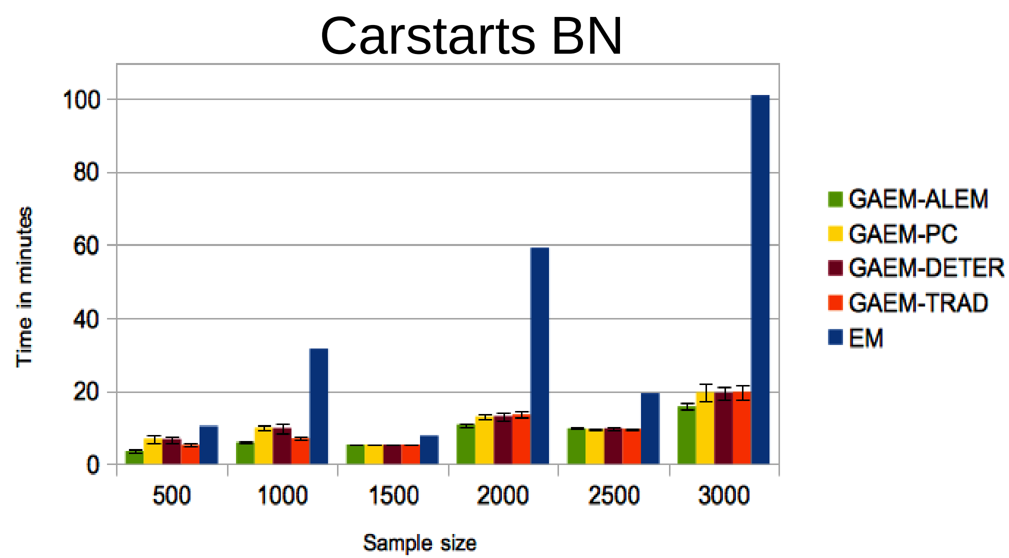
Carstarts (easy): Average number of iterations for GAEM-ALEM.

Alarm (hard): Speed-up for GAEM-ALEM relative to traditional EM.

Experiment 3: Processor Time Comparison

Carstarts BN : Pm = 0.1; Pc = 0.1; Population Size = 2; Generations = 200
Alarm BN : Pm = 0.05; Pc = 0.5; Population Size = 4; Generations =100
Traditional EM: 400 EM runs





GAEM-ALEM produces the highest speed-up for Carstarts and Alarm BNs.

Outline

- Expectation Maximization (EM) Review
- Challenges of EM and Current EM Approaches
- Our GAEM Approach
- GAEM Replacement Methods
- Experimental Results
- Discussion and Future Work

Related Work: Some EM Variants

- Problem of Local Maxima
  - EM with GAs [Jank, 2006]
  - Impact of local maxima [Wang & Zhang, 2006]
  - Random swap EM algorithm [Zhao et al., 2012]
- Problem of Time Consumption
  - Upper bound on Log-Likelihood [Zhang et al., 2008]
  - Age-layered EM method [Saluja et al., 2012]
  - Age-layered EM using MapReduce [Reed et al.,2012]

Evolutionary EM and Other EM Variants

Goal: Address one or more of the three challenges of EM

(1) EM Wrapped using Evolutionary Techniques	(2) EM Variants that Modify Original EM Algorithm
Do not modify the original EM algorithm	Modify the original EM algorithm
Do not add to the complexity	Add complexity

The GAEM method

(3) Hybrid EM Variants: (1) + (2)

Conclusion and Future Work

GAEM

- The GAEM algorithm achieves *better solution quality (in terms of LL)* in most cases.
- GAEM-ALEM produced *a speed-up of 1.5x to 7x*.

Future work

- Explore other *evolutionary and replacement strategies* – inspired by visualizations.
- Extend GAEM to *distributed computing* environments (hybrid).
- Study other ways of characterizing and using the structure of the *BN parameter search space* .

Thanks for your attention!

Questions?